Manipulationssicherheit von grafikfähigen Taschenrechnern

Robin Meis

Nörvenich, 20. September 2015

Alle verwendeten Firmen-, Markennamen und Warenzeichen sind Eigentum der jeweiligen Inhaber und dienen lediglich zur Identifikation und Beschreibung der Produkte und Dienstleistungen.

1 Vorwort

Am 10. April 2014 gab das Ministerium für Schule und Weiterbildung NRW bekannt, dass mit dem Zentralabitur 2017 erstmalig Aufgaben, die mit einem grafikfähigen Taschenrechner zu lösen sind eingeführt werden. Dadurch wurde die Anschaffung von entsprechenden Taschenrechnern ab dem Schuljahr 2014/15 verpflichtend.

Kritisiert wurden im Vorfeld vor allem die hohen Anschaffungskosten und der im Vergleich zu einem Smartphone oder Tablet relativ geringe Nutzen. Der Erlass sieht daher vor, auch die Benutzung von Tablets und Laptops zuzulassen, diese müssen allerdings in Prüfungssituationen bestimmte Voraussetzungen erfüllen, wie beispielsweise eine deaktivierte Internetkonnektivität. Um die Manipulationssicherheit der Geräte zu gewährleisten, müssen diese in einer Prüfung von der Schule gestellt werden.[1]

Die Bereitstellung eines Klassensatzes bzw. im Zentralabitur einer möglicherweise deutlich größeren Anzahl von Tablets oder Laptops stellt eine hohe finanzielle Belastung für die Schulen dar, sodass häufig auf die von den Schülern anzuschaffenden grafikfähigen Taschenrechner gesetzt wird. Diese lassen sich anders als schülereigene Laptops oder Tablets vor der Prüfung in einen definierten Zustand zurücksetzen, sodass die Schule keine eigenen Geräte bereithalten muss.

Ich selbst stehe der verpflichtenden Anschaffung von grafikfähigen Taschenrechnern kritisch gegenüber. Selbstverständlich bieten sie zahlreiche Möglichkeiten sehr spezielle Aufgabenstellungen schnell zu lösen, jedoch ist die Bedienung und die Verarbeitungsgeschwindigkeit eher mäßig. Als Nutzer eines CASIO fx-CG 20 stört mich vor allem die Darstellung von Graphen, die in einigen Situationen unpräzise ist und das Hineinzoomen mit einem großem Zeitaufwand verbunden ist. Konkurrenzanwendungen auf Tablets und Smartphones haben sich in meinen Tests als deutlich leistungsfähiger und schneller herausgestellt.

Da ich mich selber seit einigen Jahren mit Programmierung und Mikrocontrollern beschäftige, sind mir mögliche Schwachstellen in der Konzeption von manipulationssicheren Systemen bekannt, unabhängig davon, ob es sich um schuleigene Geräte wie Tablets oder Laptops in einer Prüfung oder um den Grafikrechner fx-CG 20 handelt. Ziel dieser Ausarbeitung ist es die Manipulationssicherheit von Grafikrechnern exemplarisch an dem Taschenrechner fx-CG 20 zu widerlegen.

Selbstverständlich habe ich möglichen Vorwürfen einer gefälschten Ausarbeitung Sorge getragen, sodass ich mich dazu bereit erkläre meine Lösung vorzuführen.

Inhaltsverzeichnis

1 Vorwort **2**

2 Einleitung **5**

3 Grundlagen **6**
3.1 Manipulationsmöglichkeiten . . . 6
3.1.1 Firmwaremanipulation . . . 6
3.1.2 Täuschung . . . 6
3.1.3 Wiederherstellung der gelöschten Dateien . . . 6
3.2 Abschließende Bewertung . . . 7
3.3 Auswahl der Manipulationsmöglichkeit . . . 7

4 Wiederherstellung der gelöschten Dateien **8**
4.1 Schnittstellen . . . 8
4.1.1 USB . . . 8
4.1.2 3PIN . . . 8

5 Ermittlung des Protokolls **9**
5.1 Mitschnitt des Protokolls . . . 9
5.1.1 Hardware . . . 9
5.1.2 Erklärung der Darstellung der Aufzeichnungen . . . 10
5.1.3 Mitschnitt mit 9600 Baud . . . 10
5.1.4 Mitschnitt mit 115200 Baud . . . 10
5.1.5 Zusammenfassung der Ergebnisse . . . 11
5.2 Entwicklung eines Emulators . . . 11
5.2.1 Vorgehen . . . 11
5.2.2 Programmiersprache . . . 11
5.2.3 Reihenfolge der übertragenen Datenpakete . . . 12
5.2.4 Implementierung . . . 12
5.2.5 Probleme . . . 12
5.2.6 Lösung . . . 12
5.2.7 Ausblick . . . 13
5.3 Zusammenfassung . . . 13

6 Entwicklung eines Hardwaremoduls zur Manipulation **14**
6.1 Planung . . . 14
6.1.1 Anforderungen . . . 14
6.1.2 Arbeitsschritte . . . 14
6.2 Mikrocontroller . . . 14
6.2.1 Erforderliche Bauteile . . . 14
6.2.2 Mikrocontroller-Familien . . . 14
6.2.3 AVR Grundlagen . . . 15
6.2.4 Taktfrequenz . . . 15
6.2.5 Auswahl eines Mikrocontrollers . . . 15
6.2.6 Schaltung . . . 16
6.2.7 Bewertung . . . 16
6.3 Software . . . 16
6.3.1 Erweiterung des Emulators . . . 17
6.3.2 Externe Bibliotheken . . . 17
6.3.3 Funktionsweise . . . 17
6.3.4 Ändern der Formeldatei . . . 17
6.4 Assistent zum Ändern der Formel . . . 18
6.4.1 Anforderungen . . . 18
6.4.2 Programmierung des Mikrocontrollers . . . 18

6.4.3 Adapter . . . 19
6.4.4 Assistent . . . 19

7 Bedienungsanleitung 20
7.1 Erstellen einer Formel . . . 20
7.2 Übertragung der Formel an den Computer . . . 20
7.3 Übertragung der Formel an den Mikrocontroller . . . 21
7.4 Reset des Taschenrechners . . . 21
7.5 Wiederherstellung der Formel . . . 21

8 Zusammenfassung 22

9 Ausblick 23
9.1 Verbesserung des Moduls . . . 23
9.2 Risiken für Schulen . . . 23
9.2.1 Rechtliche Bewertung . . . 23
9.2.2 Maßnahmen für Schulen . . . 23
9.2.3 Maßnahmen für Hersteller . . . 24

10 Erweiterte eigene Position 25
10.1 Erfahrungen . . . 25
10.2 Fähigkeiten der Schüler . . . 25
10.3 Weitere Taschenrechner . . . 25

11 Rechtliche Situation 26
11.1 Entwicklung eines Moduls zur Manipulation . . . 26
11.2 Veröffentlichung von Details . . . 26
11.3 Verkauf eines Moduls zur Manipulation . . . 26
11.4 Verwendung in einer Klausur . . . 26

12 Anhang 27
12.1 Datenblätter & Bedienungsanleitungen . . . 27
12.1.1 Auszug Bedienungsanleitung[5] Seite 12 . . . 27
12.2 Protokollmitschnitt . . . 28
12.2.1 9600 Baud . . . 28
12.2.2 115200 Baud . . . 29
12.3 Ablauf der Kommunikation . . . 30
12.3.1 9600 Baud . . . 30
12.3.2 115200 Baud . . . 31
12.4 Overhead . . . 32
12.5 Abbildungen . . . 33
12.5.1 Klinkenstecker . . . 33
12.5.2 Mitschnitt des Protokolls . . . 35
12.6 Modul zur Manipulation . . . 36

13 Literaturverzeichnis 38

14 Impressum 39

2 Einleitung

In Schulen eingesetzte, grafikfähige Taschenrechner gelten derzeit als Manipulationssicher. Die Taschenrechner verfügen über zahlreiche Funktionen wie einen Formelspeicher und sind teilweise programmierbar. Aus diesem Grund werden Sie vor einer Prüfung durch den Lehrer auf ihren Auslieferungszustand zurückgesetzt (Reset), sodass der gesamte Speicher gelöscht wird. Die Verwendung nachträglich installierter Anwendungen oder auf dem Taschenrechner gespeicherter Notizen wird somit unterbunden, sodass eine kontrollierte und einheitliche Prüfungssituation gewährleistet ist.

Viele der eingesetzten Taschenrechner bieten Schnittstellen zur Kommunikation mit einem Computer oder mit einem weiteren Taschenrechner. So verfügt der fx-CG 20 (auch PRIZM genannt) sowohl über einen USB Anschluss, als auch über einen dreipoligen 2.5mm Klinkenstecker, der von CASIO offiziell als 3PIN bezeichnet wird. Der USB Anschluss dient in erster Linie zur Übertragung von Anwendungen und Updates auf den Taschenrechner. Der Klinkenstecker dient dagegen zum Austausch von Dateien zwischen zwei Taschenrechnern, die über ein mitgeliefertes Kabel verbunden werden können.

Ziel dieser Ausarbeitung ist es exemplarisch an dem Modell CASIO fx-CG 20 nachzuweisen, dass grafikfähige Taschenrechner keineswegs Manipulationssicher sind.

Dazu werden zunächst die verschiedenen Möglichkeiten einer Manipulation beleuchtet und unter den Anforderungen der einfachen Umsetzbarkeit und einer möglichst geringen Wahrscheinlichkeit, dass die Manipulation durch einen Lehrer entdeckt wird bewertet. Im Anschluss wird eine dieser Möglichkeiten ausgewählt und umgesetzt.

3 Grundlagen

3.1 Manipulationsmöglichkeiten

Nachfolgend werden die Manipulationsmöglichkeiten beschrieben und die Erfüllung der Anforderungen Umsetzbarkeit und geringe Wahrscheinlichkeit der Erkennung der Manipulation durch einen Lehrer bewertet.

3.1.1 Firmwaremanipulation

Auf den ersten Blick bietet sich die Manipulation oder Veränderung der Firmware des Taschenrechners an. So ist es denkbar das durch CASIO bereitgestellte Betriebssystemupdate[2] so zu verändern, dass bestimmte Dateien beim Zurücksetzen nicht gelöscht werden und es anschließend auf den Taschenrechner aufzuspielen. Zur Umsetzung dieser Möglichkeit muss ein enormer Aufwand betrieben werden, da zunächst die Software dekompiliert oder disassembliert werden muss. Dabei wird der vorhandene Binärcode zurück in für den Menschen interpretierbaren, jedoch keineswegs lesbaren Code umgewandelt. Eine weitere Hürde stellt der Prozessor des Taschenrechners dar. Der exklusiv für CASIO hergestellte Chip SH7305 der japanischen Firma Renesas Electronics ist nicht öffentlich dokumentiert[3], wodurch eine Eigenentwicklung deutlich erschwert wird. Die Erkennung dieser Manipulation durch einen Lehrer ist aufgrund der tiefen Eingriffe in das Betriebssystem nahezu ausgeschlossen.

3.1.2 Täuschung

Eine Alternative stellt die gezielte Täuschung der Lehrkraft dar, die den Taschenrechner zurücksetzt. So ist es möglich eine eigene Anwendung zu entwickeln, die das Verhalten des originalen Menüs zum Reset des Taschenrechners nachempfindet. Da der Taschenrechner die zuletzt geöffnete Anwendung auch nach dem Ausschalten und erneuten Einschalten geöffnet hält, könnte der Dialog dem Lehrer bereits beim Einschalten gezeigt werden. An dieser Stelle würde er glauben, sich im entsprechenden Menü zu befinden und den Taschenrechner scheinbar zurückzusetzen. Unter Verwendung der inoffiziellen GCC Bibliothek PrizmSDK[4] können eigene Anwendungen für den fx-CG 20 entwickelt werden. Die Bibliothek bietet dabei auch Zugriff auf die original Menüstruktur des Taschenrechners und ermöglicht somit die Entwicklung von Anwendungen, die sich im Aussehen und Verhalten an den original CASIO Anwendungen orientieren. Diese Methode ist leicht umsetzbar, sie kann jedoch auch leicht erkannt und verhindert werden, indem der Lehrer vor dem Zurücksetzen in das Hauptmenü des Rechners wechselt und von dort aus die Einstellungen aufruft.

3.1.3 Wiederherstellung der gelöschten Dateien

Wie bereits einleitend erwähnt, verfügen viele grafikfähige Taschenrechner über Datenschnittstellen. Über solche Schnittstellen verfügt auch der fx-CG 20. So ist es denkbar, dass über diese Schnittstellen Daten an den Taschenrechner übertragen werden. Der Reset würde in diesem Fall normal ablaufen, es würden auch alle Daten gelöscht werden, allerdings würden die Dateien über einen externen Mikrocontroller unter Verwendung einer dieser Schnittstellen wieder auf den Taschenrechner aufgespielt. Mikrocontroller sind kleine Bausteine für elektronische Schaltungen, die beliebig programmiert werden können. Die USB Schnittstelle ließe sich für eine solche Anwendung prinzipiell nutzen, allerdings ist nur eine sehr kleine Anzahl von Mikrocontrollern mit dem dafür notwendigen USB Hostcontroller ausgestattet, der dazu in der Lage ist, die Rolle eines Computers einzunehmen. Damit bleibt die Verwendung des Klinkensteckers als Möglichkeit, um Dateien wie Formeln nach dem Reset an den Taschenrechner zu senden. Dafür muss ermittelt werden, welcher Übertragungsstandard verwendet wird und davon ausgehend eine Übertragung an einen Taschenrechner simuliert werden. Sofern der Taschenrechner einen von einem Mikrocontroller unterstützen Standard verwendet, ist eine Umsetzung machbar. Zugleich kann eine solche Lösung nicht im Vorfeld einer Prüfung durch einem Lehrer erkannt werden.

Da der Dialog zum Senden und Empfangen von Dateien (Menüpunkt Link im Hauptmenü) fest im System des Taschenrechners integriert ist, bleibt er auch nach einem Reset erhalten und kann für die Wiederherstellung genutzt werden.

3.2 Abschließende Bewertung

Die Möglichkeit der Firmwaremanipulation erfüllt die Anforderungen aufgrund des Aufwands und der Gefahr die Funktionsfähigkeit und Gewährleistung des Taschenrechners zu verlieren nicht.

Die Entwicklung einer täuschenden Anwendung ist zwar leicht möglich, jedoch ist sie durch den Lehrer leicht zu erkennen und bringt somit nur bedingt einen Nutzen, sodass die Anforderung der geringen Wahrscheinlichkeit die Manipulation zu entdecken nicht erfüllt wird.

Die Wiederherstellung der gelöschten Dateien über einen Mikrocontroller erfüllt alle spezifizierten Anforderungen und ist umsetzbar, sofern ein leicht zu verwendender Übertragungsstandard eingesetzt wird.

3.3 Auswahl der Manipulationsmöglichkeit

Nach erneuter Betrachtung aller Vor- und Nachteile scheint die Wiederherstellung der gelöschten Daten mithilfe eines Mikrocontrollers die beste Manipulationsmöglichkeit zu sein, sodass diese zunächst als Ziel gesetzt wird. Sollte sie sich als nicht umsetzbar herausstellen, wird die Möglichkeit der Täuschung umgesetzt.

4 Wiederherstellung der gelöschten Dateien

Ziel dieses Arbeitsabschnitts ist es einen Weg zu finden, Dateien aus dem Speicher des Taschenrechners auf einen weiteren Speicher zu übertragen, der die Löschung sämtlicher Dateien infolge eines Resets erlaubt, jedoch nach dem Reset eine Wiederherstellung der Dateien auf dem Taschenrechner ermöglicht, indem die Dateien wieder zurück auf den Taschenrechner übertragen werden.

Dazu werden zunächst die beiden Schnittstellen des Taschenrechners, USB und ein nicht näher spezifizierter Klinkenstecker im Bezug auf Ihren Nutzen für das Projekt analysiert. Im Anschluss werden erste Kommunikationstests in Verbindung mit einem Computer durchgeführt, die zum Verständnis des Übertragungsprotokolls beitragen sollen. Zusätzlich wird das Protokoll so weit wie möglich spezifiziert.

4.1 Schnittstellen

4.1.1 USB

Der USB Anschluss des Taschenrechners erfüllt mehrere Zwecke. Der jeweils gewünschte Verwendungszweck kann nach der Verbindung mit einem Computer am Taschenrechner ausgewählt werden.

Die Verwendung als USB-Massenspeicher ermöglicht den Austausch von Daten zwischen einem Computer und dem Taschenrechner ähnlich wie bei einem USB Stick. Auf diese Weise lassen sich neue Anwendungen installieren oder Dateien wie Formeln und Tabellen von einem Computer übertragen. Es ist naheliegend diesen standardisierten Anschluss zur Wiederherstellung der Daten zu verwenden. Allerdings ist die Verbindung von USB mit einem Mikrocontroller nur schwer umsetzbar, da nicht jeder Mikrocontroller den dafür notwendigen USB Host enthält. Zudem sind entsprechende Controller meist relativ teuer, haben einen großen Platzbedarf oder sind nur in SMD Ausführung lieferbar und somit nur auf speziell angefertigten Platinen verlötbar.

Zudem werden verschiedene Varianten der Bildschirmübertragung angeboten, die jedoch für diese Ausarbeitung nicht von Relevanz sind und daher nicht näher konkretisiert werden.

4.1.2 3PIN

Bei dem von CASIO als 3PIN bezeichneten Anschluss handelt es sich um einen 2.5mm Klinkenstecker, ähnlich wie bei einem Kopfhörer. Mithilfe dieses Steckers können zwei CASIO Taschenrechner über das mitgelieferte Kabel miteinander verbunden werden, um Dateien miteinander auszutauschen. Die dafür erforderliche Anwendung Link ist im Hauptmenü standardmäßig installiert und wird durch einen Reset nicht gelöscht. Genaue Spezifikationen zu diesem Stecker und dem entsprechenden Datenprotokoll wollte CASIO auf Nachfrage nicht mitteilen.

In einer offiziellen Bedienungsanleitung in englischer Sprache[5] befinden sich auf Seite 12 wenige Details zu dem verwendeten Übertragungsstandard. Wie dem Auszug unter 12.1.1 auf Seite 27 zu entnehmen ist, werden die Daten über eine serielle Verbindung übertragen. Die Angabe von Baudrate, Parität und Stopbits lässt auf eine UART oder RS-232 Verbindung schließen. Eine Spannungsmessung mit einem Multimeter und einem Oszilloskop ergab, dass 3 Volt Logikpegel (LVTTL Pegel) verwendet werden. Da RS-232 mit Pegeln zwischen ±3 Volt und ±15 Volt arbeitet, muss es sich bei dieser Verbindung um einen UART (Universal Asynchronous Receiver Transmitter) handeln.

Eine Abbildung der in den Taschenrechner eingebauten Buchse und des am mitgelieferten Kabels angebrachten Steckers befindet sich in Abbildung 1 auf Seite 33. Das Kabel verfügt an beiden Enden über jeweils einen Klinkenstecker.

5 Ermittlung des Protokolls

Da nun bekannt ist, wie zwei Taschenrechner Daten austauschen, aber nicht welche Daten sie austauschen, muss als Nächstes das Übertragungsprotokoll ermittelt werden, um zu verstehen, wie Dateien Übertragen werden und eine identische Implementierung mit einem Mikrocontroller durchführen zu können.

Bei CASIO Taschenrechnern wird grundsätzlich zwischen dem Taschenrechner der eine Datei sendet, und dem Taschenrechner, der eine Datei empfängt unterschieden. Zur besseren Übersicht wird der sendende Taschenrechner in dieser Abhandlung Master und der Empfangene Taschenrechner Slave genannt.

Ältere CASIO Modelle nutzen ein einfaches Protokoll, welches auf 9600 Baud basiert. Dabei nimmt zunächst der Master Kontakt mit dem Slave auf, indem er Hexadezimal 16 sendet. Ist der Slave bereit zum Empfangen, so schickt er Hexadezimal 13 zurück und erhält anschließend vom Master ein erstes Datenpaket. Nach der Verarbeitung schickt der Slave Hexadezimal 6, und wiederholt dies nach jedem Datenpaket, bis die gesamte Datei übertragen wurde. Ein Test ergab, dass über dieses Protokoll nicht alle Dateitypen gesendet werden können. Der fx-CG 20 ist abwärtskompatibel zu diesem Protokoll, unterstützt jedoch auch ein weiteres, derzeit unbekanntes Protokoll. So ist beispielsweise die Übertragung von Formeln nur über das aktuelle, deutlich komplexere Protokoll möglich, sodass nur dessen Verwendung in Frage kommt. Dazu muss das Protokoll zunächst aufgezeichnet, analysiert und verstanden werden.

Bei der Einheit Baud handelt es sich um die Übertragungsgeschwindigkeit des UARTs (Baudrate).

5.1 Mitschnitt des Protokolls

Um zu verstehen, welche Daten vor und während der Übertragung ausgetauscht werden, wurde die Kommunikation zwischen zwei fx-CG 20 mitgeschnitten.

5.1.1 Hardware

Um eine geeignete Komponente zum Mitschnitt der Übertragung entwickeln zu können, ist zunächst eine genaue Betrachtung des Kabels zur Verbindung zweier Taschenrechner erforderlich. Eine Messung mit einem Durchgangsprüfer ergab, dass Kontakt 1 auf beiden Seiten des mitgelieferten Kabels verbunden ist. Kontakt 2 ist mit Kontakt 3 bzw. Kontakt 3 mit Kontakt 2 gekreuzt. Daraus lässt sich schließen, dass Kontakt 1 der Massekontakt ist und Kontakt 2 und 3 die beiden Datenleitungen. Gekreuzte Datenleitungen werden auch Nullmodem Kabel genannt. Die Nummerierung der Kontakte und ihre Belegung ist in Abbildung 2 auf Seite 34 zu sehen.

Zudem ist es wichtig die Grundlagen des UARTs zu kennen. Bei zwei, gleichberechtigten Geräten, wie den beiden Taschenrechnern, werden die beiden Datenleitungen gekreuzt. In diesem Fall gibt es zwei Typen von Datenleitungen: RX und TX. RX steht für Receive (Empfangen), TX für Transceive (Senden). TX (Senden) von Taschenrechner 1 wird durch das gekreuzte Kabel mit RX (Empfangen) von Taschenrechner 2 verbunden und invers. Dadurch kann Taschenrechner 2 die gesendeten Daten von Taschenrechner 1 empfangen und umgekehrt. Der Empfangseingang (RX) ist hochohmig und belastet den Stromkreis somit kaum.

Dieses Verhalten nutzt auch die zum Mitschnitt entwickelte Hardware aus. Über einen Wandler von UART zu USB und umgekehrt, in diesem Fall ein CP2012 Breakout Board, können UART Signale mit einem Computer ausgelesen und gesendet werden. Wie im Schaltplan zu sehen, werden zwei dieser Module für den Mitschnitt benötigt. Wichtig ist hierbei, dass beide Module nur mit RX sowie der gemeinsamen Masse verbunden sind. Da der RX Eingang hochohmig ist, kann er an eine bestehende Datenleitung angeschlossen werden. Dafür muss das mitgelieferte Kabel aufgetrennt werden.

Die in Abbildung 4 auf Seite 35 sichtbare Platine erfüllt diesen Zweck. Die drei Buchsenleisten sind dafür vorgesehen, dass die Breakout Module direkt an der jeweiligen Stelle eingesteckt werden können.

Zunächst sind dabei nur die beiden äußeren Buchsenleisten interessant. Dort sind, wie im Schaltplan in Abbildung 3 auf Seite 35 zu sehen nur jeweils die RX Pins des Breakout Boards an jeweils eine Datenleitung angeschlossen, wobei das Kabel mit beiden Anschlüssen erneut verbunden wurde, sodass eine Parallelschaltung entsteht. Beide Datenleitungen können gleichzeitig abgehört werden, während zwei Taschenrechner Daten austauschen.

Die mittlere Buchsenleiste ist dagegen sowohl mit RX, als auch mit TX verbunden. Dieser Anschluss wird benötigt, wenn der Computer mit einem Taschenrechner kommunizieren soll. Dabei ist zu beachten, dass nur ein Taschenrechner angeschlossen werden darf und die gekreuzte Leitung (in Abbildung 4 auf Seite 35 mit einem Punkt markiert) zum Anschluss an den Taschenrechner verwendet werden muss.

5.1.2 Erklärung der Darstellung der Aufzeichnungen

Es existieren insgesamt vier Aufzeichnungen. Zwei davon beinhalten die aufgezeichneten Werte für 9600 Baud in 12.2.1 auf Seite 28, die anderen zwei beinhalten die aufgezeichneten Werte für 115200 Baud in 12.2.2 auf Seite 29. Die Blöcke zu den jeweiligen Baudraten sind jeweils durch eine Leerzeile getrennt und am Neubeginn der Nummerierung zu erkennen. Der obere Block entspricht jeweils den Daten, die von dem empfangenden Taschenrechner gesendet wurden, der untere Block jeweils den Daten des sendenden Taschenrechners.

Die Aufzeichnung ist von links nach rechts wie folgt aufgebaut: Zeilennummer (Hexadezimal), zwei Blöcken aus jeweils acht Bytes (Hexadezimal) und der ASCII Entsprechung. Bei ASCII Zeichen handelt es sich um eine standardisierte Entsprechung von Buchstaben, Zahlen und Sonderzeichen zu ihrem binären Wert, der beispielsweise bei Datenübertragungen übermittelt wird.

5.1.3 Mitschnitt mit 9600 Baud

Nach der Verbindung zweier Taschenrechner über die entwickelte Platine, wurde erstmalig die Übertragung einer Datei gestartet und die beiden Datenleitungen abgehört. Als Teminalprogramm kam dabei Moserial Terminal 3.0.0 unter Linux Mint 17.1 Cinnamon Edition zum Einsatz. Beide CP2012 Module wurden mit dem Computer verbunden und mit jeweils einer Moserial Instanz geöffnet. Dabei wurden die Übertragungseinstellungen wie dem Auszug aus der englischsprachigen Bedienungsanleitung 12.1.1 auf Seite 27 gemäß der Angaben für 9600 gesetzt. Nach erfolgreicher Übertragung ergab sich in der hexadezimalen Darstellung mit ASCII Entsprechungen die in 12.2.1 auf Seite 28 zu sehende Aufzeichnung.

Bei der übertragenen Datei handelt es sich um eine Formeldatei (Menüpunkt 3 eActivity) mit dem Namen ABC.g3e und den Buchstaben von A bis Z in Großschreibweise. Wie der Aufzeichnung zu entnehmen ist, wurde diese Datei bisher jedoch nicht nicht übertragen.

Nach genauer Betrachtung des ASCII Teils kristallisiert sich heraus, dass eine neue Übertragungsgeschwindigkeit von 115200 Baud im Paritätsmodus EVEN genutzt wird. 115200 und EVEN sind typische Stichworte bei UART und RS-232 Verbindungen. Die blau hervorgehobenen Bereiche markieren diese Stelle. Nach der Übermittlung der nun zu verwendenden Übertragungsgeschwindigkeit (Baudrate) und Parität, schalten beide Taschenrechner auf die neue Geschwindigkeit um, weshalb alle anschließend aufgezeichneten Daten (rot hervorgehoben) inkorrekt sind. Diese Daten hätten mit 115200 Baud gelesen werden müssen. Mit dieser Erkenntnis lässt sich auch die Angabe von 115200 Baud in der Bedienungsanleitung in 12.2.2 auf Seite 29 erklären.

5.1.4 Mitschnitt mit 115200 Baud

Im Anschluss wurde eine zweite Aufzeichnung gestartet, deren Einstellungen auf die Angaben aus der Bedienungsanleitung in 12.1.1 auf Seite 27 unter 115200 Baud basieren. Es wurde die selbe Datei erneut gesendet, sodass sichergestellt ist, dass die aufgezeichneten Daten sowohl für 9600 Baud, als auch für 115200 Baud zusammenpassen. Der Aufzeichnung in 12.2.2 auf Seite 29 sind zurzeit der Prozessor (orange), die Firmware Version (hellblau), die Dateigröße (magenta), der Dateiname (grün), der Dateityp (violett) und der Dateiinhalt (dunkelblau) zu entnehmen. Der rot

hervorgehobene Inhalt der ersten Zeile ist fehlerhaft und liegt in der vorherigen Kommunikation mit 9600 Baud begründet, die nicht gleichzeitig aufgezeichnet wurde.

Die Dateigröße wird im Mitschnitt in der ASCII Entsprechung als D8 angeben, wobei es sich um die hexadezimale Darstellung der Dateigröße von umgerechnet 216 Bytes handelt.

Das Ende der Übertragung wird mit Hexadezimal 18 mitgeteilt.

5.1.5 Zusammenfassung der Ergebnisse

Nach ersten Erkenntnissen handeln die Taschenrechner die zu verwendende Übertragungsgeschwindigkeit vor der Übertragung aus, sodass ältere CASIO Taschenrechner Dateien mit einer Übertragungsgeschwindigkeit von 9600 Baud übertragen und neuere Modelle eine Geschwindigkeit von 115200 Baud nutzen.

Die eigentliche Datei macht nur einen kleinen Teil der gesamten Kommunikation zwischen den Taschenrechnern aus. Zudem werden zahlreiche Informationen übertragen, deren Bedeutung derzeit unbekannt ist.

Die Dateigröße von 216 Bytes scheint sehr groß zu sein. Der Dateiinhalt wird in 8 Bit Standard ASCII Zeichen übertragen. Da die Datei 26 Zeichen enthält, müsste sie 26 * 8 Bit = 26 Byte groß sein, sodass davon auszugehen ist, dass die Datei zusätzliche Informationen enthält.

5.2 Entwicklung eines Emulators

Ein Emulator bildet das Verhalten einer bestimmten Hardware nach, sodass Software auch mit anderer Hardware als derer, für die die Anwendung ursprünglich entwickelt wurde zusammenarbeitet.

Auf den konkreten Anwendungsfall bezogen bedeutet dies, dass zunächst ein Emulator entwickelt wird, der auf einem Computer eingesetzt wird, um das Grundverhalten des Protokolls zu verstehen. Derzeit ist bekannt, welche Daten bei der Kommunikation gesendet werden, jedoch nicht, wann sie gesendet werden. Um eine Dateiübertragung zu ermöglichen, muss daher zunächst verstanden werden, in welcher Reihenfolge die Informationen übertragen werden. Anschließend können die Informationen dokumentiert und in eine Anwendung für einen Mikrocontroller umgesetzt werden.

Ziel ist es anhand des vorliegenden Protokollmitschnitts den Empfang von Dateien des Taschenrechners zu ermöglichen.

5.2.1 Vorgehen

Zunächst wird eine Anwendung auf einem Computer entwickelt, die über ein CP2012 Breakout Board mit dem Taschenrechner kommuniziert und das Verhalten eines empfangenden Taschenrechners emuliert. Dabei kommt die mittlere Buchsenleiste der in 5.1.1 auf Seite 9 entwickelten Hardware zum Einsatz.

Es wird schrittweise versucht den Zusammenhang der gesendeten und empfangenen Datenpakete herzustellen, sodass konkretisiert werden kann, in welcher Reihenfolge die Daten übertragen werden.

5.2.2 Programmiersprache

Der Emulator dient zu Testzwecken und zur Ermittlung des Protokolls. In diesem Stadium werden besondere Ansprüche an eine schnelle und unkomplizierte Umsetzung, sowie eine verständliche Syntax gesetzt. Die Ausführungsgeschwindigkeit spielt bei der Entwicklung des Emulators eine geringe Rolle, da die Übertragungsgeschwindigkeit sehr gering ist.

Um den genannten Anforderungen gerecht zu werden, wird die Scriptsprache Python3 mit dem Modul pySerial verwendet, dass es ermöglicht direkt auf die serielle Schnittstelle des CP2012 zuzugreifen. Eine möglicherweise erforderliche Portierung auf die Programmiersprache C, um eine geeignete Anwendung für einen Mikrocontroller zu entwickeln wird dabei in Kauf genommen.

5.2.3 Reihenfolge der übertragenen Datenpakete

Der Ablauf der Kommunikation wurde in 12.3.1 auf Seite 30 und in 12.3.2 auf Seite 31 dokumentiert. Dabei handelt es sich um die selben aufgezeichneten Daten wie in 12.2.1 auf Seite 28 und in 12.2.2 auf Seite 29. Jeder Farbwechsel kennzeichnet dabei ein gesendetes Datenpaket, nach dessen Empfang der jeweils andere Taschenrechner eine Antwort sendet. Die rot hervorgehobenen Bereiche kennzeichnen fehlerhafte Daten aufgrund des Baudratenwechsels.

Zur Ermittlung der Reihenfolge wurden in mehreren Schritten Bytes zu den zu sendenden Datenpaketen hinzugefügt und auf Antworten des Taschenrechners gewartet. Schlägt die Kommunikation fehl, ohne dass eine Antwort gesendet wurde, so wurden zu wenig Bytes gesendet, schlägt die Kommunikation fehl, obwohl eine Antwort gesendet wurde, sind neue Datenpakete übermittelt worden, auf die noch nicht geantwortet werden konnte, da eine entsprechende Implementierung noch aussteht.

Zu diesem Zeitpunkt ist die Kommunikationsreihenfolge vollständig ermittelt und der Emulator funktionsfähig.

5.2.4 Implementierung

Der Quellcode des Emulators ist aufgrund von Platzgründen nicht eingebunden. Die Implementierung orientiert sich an der ermittelten Übertragungsreihenfolge und antwortet auf erwartete Datenpakete mit aufgezeichneten Datenpaketen. Bei dieser Methode handelt es sich um einen Replay-Angriff, bei dem bekannte Daten erneut gesendet werden.

5.2.5 Probleme

Die vollständige Bedeutung der Datenpakete konnte bisher nicht geklärt werden. Daher ist die Implementierung wahrscheinlich nicht vollständig korrekt und nicht mit den CASIO Standards kompatibel. Der Dateiempfang beliebiger Dateien ist dennoch möglich, womit das Ziel des Emulators erreicht wurde.

Beim Empfang verschiedener Dateien kristallisierte sich heraus, dass sich der in 12.4 auf Seite 32 übertragene Overhead abhängig vom Dateiinhalt verändert. Um eine Datei auf den Taschenrechner zu übertragen, muss ermittelt werden, wie dieser Overhead zu Stande kommt. Andernfalls würde ein emulierter Sender möglicherweise keine Daten senden können, da der empfangene Taschenrechner aufgrund eines Fehlers die Übertragung frühzeitig abbrechen würde.

5.2.6 Lösung

Die Erzeugung des Overheads ließ sich nicht zweifelsfrei nachvollziehen. Es bleibt unklar, ob es sich dabei um einen Übertragungstest oder eine Checksumme handelt, welche eine korrekte Übertragung gewährleisten sollen, oder ob es sich um eine verschlüsselte Form der übertragenen Datei handelt. Die verschlüsselte Form könnte genutzt werden, um die Entwicklung einer Manipulationslösung zu verhindern, und zugleich als Übertragungstest oder Checksumme dienen, da sich ihr entschlüsselter Inhalt mit dem Dateiinhalt decken müsste. Wird nur ein Bit verändert, würden die Daten nicht mehr übereinstimmen und ein Fehler bzw. eine Manipulation erkannt werden.

Bisher war lediglich geplant den Dateiinhalt nach dem Reset erneut an den Taschenrechner zu senden. Aufgrund der veränderten Umstände, muss nun auch der nicht näher bekannte Overhead aufgezeichnet, gespeichert und anschließend mit übertragen werden. Dies hat zur Folge, dass die gespeicherten Dateien größer werden. Anstatt den Overhead dynamisch und abhängig vom Dateiinhalt zu erzeugen, wird er also per Replay Angriff an den Taschenrechner gesendet.

Da der Overhead größer ist als die tatsächliche Dateigröße einer Formeldatei, kann dieser nicht den derzeit unbekannten, zusätzlichen Informationen in einer Formeldatei entsprechen.

5.2.7 Ausblick

Die Erzeugung des Overheads kann möglicherweise nachvollzogen werden, wenn die Ergebnisse mehrerer Dateiübertragungen verglichen werden. Dies wird jedoch aus Zeitgründen nicht Teil dieser Abhandlung sein.

5.3 Zusammenfassung

Im Zuge dieses Arbeitsabschnitts, konnte die Reihenfolge der gesendeten Datenpakete bestimmt werden. Dazu wurde ein Emulator entwickelt, der einen Taschenrechner im Empfangsmodus simuliert und somit übertragende Dateien von einem anderen Taschenrechner empfangen kann. Er ist kompatibel mit Formeldateien, jedoch nicht mit anderen Dateitypen wie Tabellen oder Programmen.

Die Erzeugung des Overheads konnte nicht nachgebildet werden, sodass es nicht möglich ist ihn abhängig von der Datei zu erzeugen. Stattdessen muss er mit der empfangenen Datei abgespeichert und bei der Wiederherstellung gesendet werden. Zu diesem Zweck wurde dem Empfangsemulator ein Puffer hinzugefügt, der die Datei einschließlich des Overheads aufzeichnet und nach der Übertragung in eine frei wählbare Datei schreibt. Der Inhalt des so entstandenen Dumps muss an späterer Stelle durch die Manipulationshardware wieder gesendet werden.

Die gewonnenen Erkenntnisse haben sich im Bezug auf die Entwicklung eines Emulators zum Empfang einer Datei als korrekt herausgestellt. Dennoch kann nicht abschließend geklärt werden, ob alle Ergebnisse vollständig mit dem CASIO Standard übereinstimmen. Es ist ebenfalls möglich, dass der Emulator aufgrund von Toleranzen funktioniert, die im CASIO Standard definiert sind. Der Standard konnte nicht vollständig ermittelt werden, sodass es Teile des Protokolls gibt, in denen die bisher entwickelte Implementierung Fehler aufweisen kann.

6 Entwicklung eines Hardwaremoduls zur Manipulation

Ziel dieses Arbeitsabschnitts ist es anhand der aus der Protokollanalyse gewonnenen Informationen ein Hardwaremodul zu entwickeln, die eine Formeldatei speichert und diese nach dem Reset wieder an den Taschenrechner überträgt.

6.1 Planung

Nachfolgend werden die Anforderungen an die Hardware formuliert und ein Vorgehen zur Umsetzung entwickelt. Im weiteren Verlauf wird ein Hardwarekonzept konkretisiert werden und auf dessen Basis eine Bewertung der Erfüllung der Anforderungen durchgeführt.

6.1.1 Anforderungen

Das Modul sollte eine beliebige Datei speichern und nach einem Reset über den 3PIN an den Taschenrechner übertragen. Der verwendete Speicher sollte einen möglichst hohen Speicherplatz aufweisen, und dabei in einer kleinen Bauform verfügbar sein.

Die Unterstützung eines UARTs mit den Baudraten 9600 Baud und 115200 Baud mit dem Paritätsmodus EVEN ist essentiell.

Als Versorgungsspannung sollten 3.3V eingesetzt werden, um das Modul direkt mit dem 3PIN Anschluss verbinden zu können.

Die entwickelte Hardware sollte ohne Fachwissen benutzbar sein, sodass sie prinzipiell für den Verkauf und eine breite Nutzergruppe angeboten werden könnte.

6.1.2 Arbeitsschritte

Wie bereits eingangs beschrieben, soll ein Mikrocontroller zur Speicherung der Formeldatei dienen. Dazu muss zunächst ein Mikrocontroller anhand der Kriterien ausgewählt werden. Zugleich müssen die Rahmenbedingungen Versorgungsspannung und Frequenz beachtet werden.

Im Anschluss wird ein Laboraufbau entwickelt, der erste Tests am Aufbau erlaubt und schnelle Veränderungen zulässt. Auf Basis dieses Testaufbaus wird die für den Mikrocontroller erforderliche Software entwickelt, die zur Übertragung der Datei benötigt wird.

Nach einem erfolgreichem Test von Hard- und Software wird eine Prototypenplatine entwickelt, die die Funktionsweise grundsätzlich zeigen kann.

6.2 Mikrocontroller

6.2.1 Erforderliche Bauteile

Der Baustein zur Manipulation besteht im Wesentlichen aus nur einem einzigen integrierten Schaltkreis und der erforderlichen Grundbeschaltung. Dabei handelt es sich um einen Mikrocontroller, der die Kommunikation mit dem Taschenrechner ermöglicht und zugleich als Speicher für die Formeldatei dient.

6.2.2 Mikrocontroller-Familien

Aufgrund des vorhandenen Programmiergeräts für AVRs wird ein Mikrocontroller aus der 8-Bit-Familie der Firma Atmel verwendet. Zur Auswahl stehen dabei Mikrocontroller der ATtiny und der ATmega Gruppe.

ATtinys zeichnen sich durch ihre besonders kleinen Gehäuse mit nur wenigen Pins aus. Ihre Funktionen sind im Vergleich zu den ATmegas begrenzt, sodass Bootloader Unterstützung und UART meist fehlen.

ATmegas weisen deutlich mehr Pins auf, unterstützen aber im Gegensatz zu den ATtinys auch den für das Vorhaben essentiellen UART als Bestandteil ihrer Hardware.

UART Unterstützung lässt sich grundsätzlich sowohl in Software, als auch in Hardware umsetzen, sodass ATtinys softwareseitig mit einem UART ausgestattet werden können. Jedoch ist der

Aufwand zur Implementierung deutlich größer und die verfügbaren Übertragungsgeschwindigkeiten können geringer sein. Da die Verwendung eines Hardware UART dem Prozessor Rechenlast erspart, ist dieser vor allem bei der Verwendung von Paritätsbits vorzuziehen, sodass der Fokus auf der Auswahl eines ATmegas liegt.

6.2.3 AVR Grundlagen

Bei AVRs handelt es sich um 8-Bit RISC Mikrocontroller der Firma Atmel. Die Programmierung erfolgt in der Regel mit den Programmiersprachen Assembler oder C. Teilweise werden jedoch auch andere Sprachen wie BASIC benutzt. Für dieses Projekt wird die Programmiersprache C verwendet.

Mikrocontroller verfügen über einen integrierten Programmspeicher (Flash Speicher) und Arbeitsspeicher (SRAM). Zusätzlich verfügen viele Controller über einen EEPROM, der genutzt wird, um Daten auch nach dem Trennen der Versorgungsspannung speichern zu können und einen integrierten Oszillator zur Takterzeugung. Der Programmspeicher kann während der Laufzeit des Programms nicht überschrieben werden, es sei denn es wird ein Bootloader verwendet. Der EEPROM kann dagegen auch während der Laufzeit des Programms beschrieben werden.

Außerdem verfügen Mikrocontroller über IO Pins, die genutzt werden können, um digitale und analoge Signale einzulesen und auszugeben. Diese Pins sind teilweise mit Sonderfunktionen wie einem Hardware I2C Bus oder einem Hardware UART belegt.

Ein Programm wird im Normalfall über einen angepassten SPI Bus auf den Controller geflasht, während er sich in der Schaltung befindet (ISP). Zudem ist es möglich ein Programm mithilfe eines Bootloaders zu flashen. In diesem Fall kann das Programm auch per UART an den Controller Übertragen werden.

Der interne Oszillator kann grundsätzlich den Mikrocontroller mit einem Systemtakt versorgen, jedoch reicht die Genauigkeit und Temperaturstabilität für viele Anwendungen nicht aus. So ist der Betrieb eines UARTs nur mit einem externen Quarz zuverlässig möglich. Soll ein externer Quarz verwendet werden, so muss dies in den Einstellungen des Controllers (Fuses) konfiguriert werden.

6.2.4 Taktfrequenz

Um den UART mit einer bestimmten Baudrate verwenden zu können, sind verschiedene Frequenzen unterschiedlich gut geeignet. Ziel ist es, eine Taktfrequenz zu finden, die sowohl für 9600 Baud, als auch für 115200 Baud geeignet ist. Je nach Frequenz sind die Fehler für die einzelnen Baudraten so hoch, dass eine Übertragung nicht mehr möglich ist.

Ideal ist der Einsatz eines Baudratenquarzes. Bei der Verwendung eines Baudratenquarzes mit einer Frequenz von 1.8432 MHz liegt sowohl der Fehler für 9600 Baud, als auch für 115200 Baud bei 0%.

Da ein entsprechender Baudratenquarz zum Zeitpunkt der Durchführung dieses Projekts jedoch nicht vorhanden war, muss ein Standardquarz verwendet werden. Die kleinste mögliche Standardfrequenz für beide Baudraten liegt bei 16 MHz, wobei es bei 9600 Baud zu einem Fehler von 0.2% und bei 115200 Baud zu einem Fehler von 3.7% kommt. Eine Übertragung bei einem Fehler von 3.7% ist grundsätzlich noch möglich, befindet sich jedoch in einem kritischen Bereich.

Die hohe Taktfrequenz erhöht den Stromverbrauch. Da es sich bei der Verwendung in einem Taschenrechner um ein batteriebetriebenes Gerät handelt, sollte die Frequenz so klein wie möglich gehalten werden. Da die Störanfälligkeit ebenfalls mit der Frequenz steigt, sollte wenn vorhanden an dieser Stelle ein 1.8432 MHz Baudratenquarz verwendet werden.

Durch die Frequenz von 16 MHz kann der Mikrocontroller nicht mit 3.3 Volt betrieben werden, sodass nur ein Betrieb mit 5 Volt möglich ist. Möglicherweise muss ein Pegelwandler eingesetzt werden.

6.2.5 Auswahl eines Mikrocontrollers

Es bietet sich eine Vielzahl von Mikrocontrollern an. Die Anforderung an einen großen Speicher impliziert in den meisten Fällen einen großen Controller mit vielen Pins. Vor allem der EEPROM

sollte möglichst groß sein, da dieser primär als Datenspeicher in Frage kommt. Oft ist der der EEPROM 512 Byte oder 1024 Byte groß. Aufgrund der ohnehin relativ großen Dateigröße in Kombination mit dem zusätzlich zu speichernden Overhead kann dieser Speicher schnell zu klein werden.

Somit bleibt nur die Verwendung des normalerweise relativ großen Programmspeichers. In ihm kann die Formeldatei als Variable im Programm abgelegt werden, sodass der große Speicher genutzt wird. Da die Variable ausschließlich aus dem Programmspeicher geladen wird, benötigt sie keinen zusätzlichen Arbeitsspeicher. Da das Programm nur eine sehr geringe Größe haben wird, sind hier keine Platzprobleme zu erwarten. Nachteil dieser Lösung ist, dass das Programm für die Übertragung einer anderen Datei neu kompiliert werden muss. Eine direkte Übertragung der Datei vom Taschenrechner auf den Mikrocontroller ist damit nur schwer möglich, sodass ein Umweg über einen Computer erforderlich wird.

Da die Anforderung eines großen EEPROMS in die Anforderung an einen ausreichend großen Programmspeicher umgewandelt wurde, spielt der EEPROM keine Rolle mehr und es stehen nahezu alle angebotenen Controller zur Wahl.

Aufgrund vorhandener Erfahrungen mit einem ATmega 8 wird dieser verwendet. Er unterstützt eine maximale Frequenz von 16 MHz und verfügt über 8 KByte Programmspeicher. Der ausgewählte Controller muss nicht die beste Wahl darstellen, er erfüllt die Anforderungen jedoch ausreichend.

6.2.6 Schaltung

Der in Abbildung 5 auf Seite 36 dargestellte Schaltplan berücksichtigt den Betrieb mit 16 MHz bei 5 Volt. Aufgrund der hohen Frequenz sind Abblockkondensatoren an den Versorgungspins unerlässlich. Bei niedrigen Frequenzen sind sie zwar empfehlenswert, jedoch nicht grundsätzlich erforderlich.

Da an der TX Leitung des Taschenrechners LVTTL (3.3 Volt) Pegel anliegen, ist es nicht auszuschließen, dass der gesamte Prozessor mit 3.3 Volt betrieben wird. Ein öffentliches Datenblatt zu dem exklusiv für CASIO hergestellten Prozessor existiert nicht, sodass nur die Datenblätter ähnlicher Prozessoren des Herstellers Renesas Electronics genutzt werden können.

Laut den dort dokumentierten Spannungsangaben sind die Eingänge für Abweichungen von ± 0.3 Volt von der Versorgungsspannung ausgelegt, sodass die Verwendung von 5 Volt Pegeln zu Schäden am Prozessor des Taschenrechners führen könnte. Zum Schutz wurde ein nahezu unbelasteter Spannungsteiler an den TX Pin des Mikrocontrollers angeschlossen, der die Spannung auf 3 Volt herabsetzt. Mit einem Gesamtwiderstand von etwa 1 kOhm fließen etwa 5 mA bei einem logisch 1 Signal am UART.

Für den RX Pin des Mikrocontrollers wird keine Verstärkerschaltung benötigt, da die Pins eines AVRs bereits bei Spannungen unter 3.3 Volt schalten.

Die Spannungsversorgung des Mikrocontrollers erfolgt über eine 9 Volt Blockbatterie, die über einen an der Platine angelöteten Batterieclip angeschlossen wird. Da der Mikrocontroller für eine Spannung von maximal 5 Volt ausgelegt ist, wandelt ein Spannungsregler die vorhandenen 9 Volt auf 5 Volt.

6.2.7 Bewertung

Die Anforderung an den Einsatz von einer Versorgungsspannung von 3.3 Volt konnte nicht erfüllt werden, da der Mikrocontroller bei eine Taktfrequenz von 16 MHz mit 5 Volt betrieben werden muss. Stattdessen wurde ein Spannungsteiler zur Pegelanpassung eingesetzt.

6.3 Software

Die Software wird an die Erkenntnisse aus der Entwicklung des Emulators angelehnt, wobei nun ein Sender anstatt eines Empfängers implementiert werden muss.

Zu Testzwecken wird die Software zunächst zur leichteren Fehlerdiagnose mit dem Emulator getestet, und erst im Anschluss mit dem Taschenrechner verbunden.

Die Software wird in C programmiert und mit dem Crosscompiler AVR-GCC kompiliert. Dieser ermöglicht es ein C Programm auf einem Computer in eine ausführbare Datei umzuwandeln, die anschließend auf den Mikrocontroller übertragen wird.

6.3.1 Erweiterung des Emulators

Bisher gibt der Emulator die übertragene Datei und ihren Overhead Hexadezimal in einem Terminal aus. Um die Datei weiterverwenden zu können, muss der Emulator so erweitert werden, dass er jedes empfangene Byte in einer Datei speichert.

Dazu wurde ein Puffer hinzugefügt, in dem jedes Byte des Overheads und der Datei bis zum Ende der Übertragung gespeichert wird. Anschließend kann der Benutzer wählen, ob die Datei gespeichert werden soll. Soll sie gespeichert werden, wird ein Dateiname abgefragt. Eine vorgegebene Dateiendung gibt es dabei nicht.

In der Datei ist jedes empfangene Byte durch ein Komma getrennt und in der C Notation für Hexadezimalzahlen gespeichert, sodass die entstandenen Daten direkt in ein Array im Programm des Mikrocontrollers eingefügt werden können.

6.3.2 Externe Bibliotheken

Aus Zeit- und Stabilitätsgründen kommt die UART Bibliothek von Peter Fleury[6] zum Einsatz. Sie gilt im Bereich der AVR Entwicklung als Standardimplementierung und erlaubt die grundlegende Benutzung der in dem Mikrocontroller integrierten UART Schnittstelle. Darauf aufbauend wird die Kommunikation mit dem Taschenrechner implementiert werden.

6.3.3 Funktionsweise

Das Programm wird einschließlich der Formeldatei kompiliert und anschließend auf den Mikrocontroller übertragen. Die Formeldatei ist dann hardcodiert und kann nur durch Änderungen im Programm verändert werden.

Nach dem Start des Mikrocontrollers sendet er laufend Hexadezimal 10. Sobald sich der Taschenrechner im Empfangsmodus befindet, antwortet dieser mit Hexadezimal 10 und die Übertragung beginnt. Nach der Übertragung beendet sich das Programm bis zum Reset oder Trennung der Stromversorgung des Mikrocontrollers. Dazu wird eine endlose Schleife aufgerufen. Mit Reset ist bei Mikrocontrollern nicht wie beim Taschenrechner die Wiederherstellung der Werkseinstellungen gemeint, sondern der Rücksprung des Programms an den Anfangspunkt (Neustart).

6.3.4 Ändern der Formeldatei

Um die im Programm des Mikrocontrollers hinterlegte Datei zu ändern, muss der Taschenrechner zunächst über den in 5.1.1 auf Seite 9 vorgestellten Adapter mit einem Computer verbunden werden. Dabei ist erst der Klinkenstecker einzustecken, und dann der USB Stecker. Andernfalls kann es zu Kurzschlüssen und Schäden am Computer oder Taschenrechner kommen.

Nun wird der in 5.2 auf Seite 11 vorgestellte Emulator zum Empfang gestartet und die gewünschte Formeldatei am Taschenrechner gesendet. Der Computer speichert die Datei einschließlich des Overheads in einer durch den Benutzer anzugebenden Datei ab.

Der Inhalt der Datei wird anschließend in das Array *data* (file.c) im Programm des Mikrocontrollers eingetragen. Zudem muss die Größe des Arrays in der Konstante *BYTES_FORMEL* in den Dateien file.h und main.h angepasst werden, die die Anzahl der Elemente im Array bereithält.

Im Anschluss wird der Taschenrechner vom Computer getrennt. Dazu muss zunächst die USB Verbindung und anschließend der Klinkenstecker getrennt werden.

Im letzten Schritt muss der Mikrocontroller über ein ISP Programmiergerät mit dem Computer verbunden werden. Ein Skript kompiliert und überträgt das Programm automatisch. Jetzt ist der Mikrocontroller einsatzbereit und kann wieder mit dem Taschenrechner verbunden werden.

6.4 Assistent zum Ändern der Formel

Die in 6.3.4 auf Seite 17 Seite vorgestellte Methode zum Ändern der Formeldatei ist sehr zuverlässig, jedoch auch sehr aufwändig und kompliziert. Ziel ist es nun einen Assistenten zu entwickeln, der diesen Arbeitsschritt vereinfacht und automatisiert, sodass er von jedem Anwender durchgeführt werden kann.

6.4.1 Anforderungen

Der Benutzer soll nur ein einziges Programm starten müssen. Diese sollte ihn durch den gesamten Vorgang der Übertragung begleiten. Des weiteren soll der Aufwand des manuellen Einfügens der Formel in den Programmcode und des Kompilierens entfallen.

Die Übertragung der Formel vom Taschenrechner an den Computer soll mit einem leicht zu verwendenden Adapter erfolgen. Die Verbindung soll dabei direkt über das beim Taschenrechner mitgelieferte Klinkenkabel erfolgen oder über ein eigenes Kabel mit einem Klinkenstecker verfügen.

Die Übertragung der Formel an den Mikrocontroller soll ebenfalls möglichst einfach erfolgen. Derzeit wird dazu ein ISP Programmiergerät benötigt. Diese Geräte sind relativ teuer und benötigen einen 6 poligen Stecker auf der Mikrocontroller Platine. Stattdessen sollte ein günstiger Adapter entwickelt werden, der möglichst mit den vorhanden Anschlüssen an der Platine auskommt.

Zur Umsetzung werden zunächst die Möglichkeiten zur Programmierung des Mikrocontrollers im Bezug auf die formulierten Anforderungen analysiert. Anschließend wird ein Adapter für die Kommunikation mit dem Taschenrechner und zur Programmierung des Taschenrechners entwickelt. Die Umsetzungsmöglichkeiten der Software werden ausgehend von dem entwickelten Adapter analysiert werden.

6.4.2 Programmierung des Mikrocontrollers

Der Mikrocontroller wird derzeit über ISP programmiert. Der dabei entstehende Kostenaufwand stellt ein erhebliches Problem im Bezug auf den Gesichtspunkt der Nutzerfreundlichkeit dar.

Die Platine ist mit einem Batterieclip und einem Klinkenstecker ausgestattet. Der Batterieclip ermöglicht lediglich die Verbindung des Taschenrechners mit einer Spannungsquelle und kann nicht zu Programmierung genutzt werden. Der Klinkenstecker wird zur Kommunikation mit dem Taschenrechner verwendet. Er ist an die beiden UART Pins des Mikrocontrollers gelötet.

Unter Verwendung eines Bootloaders können Programme auch ohne ein ISP Programmiergerät auf einen Mikrocontroller geflasht (übertragen) werden. Der Bootloader wird an einer bestimmten Stelle im Programmspeicher des Mikrocontrollers abgelegt und der Mikrocontroller mit einer Fuse angewiesen von dieser Stelle zu starten. Ist die entsprechende Fuse gesetzt, kann ein Programm an dieser Stelle den gesamten Programmspeicher des Mikrocontrollers, außer den in dem es selbst gespeichert ist überschreiben.

Dabei muss der Speicher mit gültigem Programmcode beschrieben werden. Der Programmcode wird bevor er geschrieben wird über eine Schnittstelle des Mikrocontrollers an den Bootloader übertragen. Der Bootloader schreibt diese Daten dann in den Programmspeicher.

Mögliche Schnittstellen sind dabei im Fall des verwendeten ATmega 8 UART, I2C und SPI. Die UART Schnittstelle des Mikrocontrollers wird bereits über den Klinkenanschluss bereitgestellt.

Da ein Bootloader grundsätzlich vor dem Start des eigentlichen Programms ausgeführt wird, kann er auf eingehende Daten auf der UART Schnittstelle warten, die ein neues Programm enthalten. Erhält er in einem bestimmten Zeitraum keine Antwort, so startet er das bereits vorhandene Programm.

Es bietet sich an die UART Schnittstelle für dieses Vorhaben zu verwenden. Sie wird bereits über einen Klinkenstecker von der Platine herausgeführt und kann direkt an ein CP2012 Breakout Board angeschlossen werden. Der Computer kann dann das Programm über die serielle Schnittstelle an den Mikrocontroller senden.

Aus Zeitgründen kommt an dieser Stelle ein fertiges Projekt zum Einsatz. Der Bootloader KAVR[7] erfüllt alle Anforderungen, und muss einmalig über die ISP Schnittstelle auf den Controller geflasht werden. Dies würde bereits in der Produktion geschehen, sodass der Endanwender lediglich den UART Adapter benötigt.

6.4.3 Adapter

Derzeit erfolgt der Empfang der Datei mit dem in 5.1.1 auf Seite 9 vorgestellten Adapter. Er stellte für das Reverse Engineering ein unerlässliches Werkzeug dar. Der Endbenutzer benötigt jedoch die Möglichkeit des Protokollmitschnitts nicht. Zudem stellen die verschiedenen Anschlussmöglichkeiten und der fehlende Verpolungsschutz ein Risiko für den Taschenrechner dar. Ein falscher Anschluss könnte den 3PIN Anschluss oder den gesamten Taschenrechner beschädigen.

Bisher wird ein CP2012 Breakout Board verwendet, um die UART Schnittstelle des Taschenrechners über den bisherigen Adapter mit dem Computer zu verbinden. Diese Konstellation soll beibehalten werden, jedoch so umgesetzt werden, dass Anschlussfehler ausgeschlossen sind.

Dazu bietet es sich an, das Breakout Board mit einem Klinkenkabel auszustatten, dass direkt in die Klinkenbuchse des Taschenrechners gesteckt werden kann. Dabei erfolgt die Verbindung ähnlich wie bei der Verbindung zweier Taschenrechner.

Ein solcher Adapter kann nicht mit der Mikrocontroller Platine verbunden werden, da diese ebenfalls mit einem Klinkenstecker ausgestattet ist. Die Verwendung eines identischen Adapters ist jedoch sinnvoll, da sowohl zur Übertragung einer Datei an den Computer, als auch zum Programmieren des Mikrocontrollers eine UART Schnittstelle verwendet wird.

Um einen einheitlichen Adapter zu erstellen, wird anstatt eines Klinkensteckers eine Klinkenbuchse mit dem CP2012 Breakout Board verbunden. Dadurch kann die Mikrocontroller Platine direkt über den angelöteten Klinkenstecker verbunden werden. Der Taschenrechner wird dabei mithilfe des durch CASIO mitgelieferten Klinkenkabels mit der Klinkenbuchse verbunden.

6.4.4 Assistent

Der Assistent verknüpft die einzelnen Schritte miteinander und gibt zusätzlich Anweisungen an den Nutzer aus, welches Gerät wann anzuschließen ist. Der Empfang einer Datei ist bisher über einen in Python programmierten Emulator möglich. Es sind minimale Anpassungen erforderlich, weshalb des Assistent in Python3 entwickelt wird. Es wird auf eine grafische Oberfläche verzichtet. Da der Nutzer jedoch keine aufwändigen Schritte durchführen muss, stellt dies kein Problem dar.

7 Bedienungsanleitung

Der Mikrocontroller muss vor einer Prüfung mit der Formel ausgestattet werden. Dazu sind die folgenden Schritte erforderlich:

7.1 Erstellen einer Formel

1. Im Hauptmenü wird der Menüpunkt eActivity (Taste 3) ausgewählt
2. Es wird eine neue Formel mit dem Menüpunkt NEW (F2) erstellt
3. Es wird ein Name eingeben
4. Der Inhalt der Formeldatei wird erstellt
   Dieser kann Text oder Formeln in mathematischer Darstellung (z.B. Wurzelzeichen und Brüche) enthalten. Die Eingabe erfolgt dabei wie bei der Berechnung. Es stehen Großbuchstaben von A bis Z zu Verfügung.
5. Die Datei wird gespeichert
   (a) Der Menüpunkt FILE (F1) wird ausgewählt
   (b) Der Menüpunkt SAVE (F1) wird ausgewählt

7.2 Übertragung der Formel an den Computer

1. Der Assistent muss gestartet werden
2. Anschluss des Adapters
   Der Assistent wartet auf den Anschluss eines Adapters durch den Nutzer und versucht eine Verbindung aufzubauen
3. Übertragung der Formel an den Computer
   (a) Anschluss des Taschenrechners
   Der Assistent bittet den Nutzer den Taschenrechner anzuschließen und den Anschluss mit Enter zu bestätigen. Dazu wird das beim Taschenrechner mitgelieferten Klinkenkabel verwendet
   (b) Übertragung der Formeldatei
   Der Assistent erwartet den Start der Übertragung durch den Nutzer. Dazu sind am Taschenrechner folgende Schritte erforderlich:
      i. Im Hauptmenü wird der Menüpunkt Link (Taste E) ausgewählt
      ii. Sollte als Kabeltyp USB ausgewählt sein, so muss mit F4 (CABLE) der Kabeltyp auf 3PIN (F2) gesetzt werden
      iii. Die Dateiübertragung kann durchgeführt werden. Es werden nur Formeldateien durch den Mikrocontroller unterstützt
         A. Der Menüpunkt TRANSMIT (F1) wird ausgewählt
         B. Der Menüpunkt STRGMEM (F2) wird ausgewählt
         C. Der Menüpunkt SELECT (F1) wird ausgewählt
         D. Die Formeldatei wird markiert. Sie ist durch Ihren Namen erkennbar. Es können nur Dateien mit der Endung .g3e verarbeitet werden. Die Datei kann mit dem Steuerkreuz ausgewählt werden
         E. Die gewünschte Formeldatei wird mit SELECT (F1) für die Übertragung ausgewählt. Es darf nur eine Datei ausgewählt werden
         F. Die Datei wird mit dem Menüpunkt TRANSMIT (F6) übertragen
         G. Die Übertragung wird mit dem Menüpunkt Ja (F1) gestartet

7.3 Übertragung der Formel an den Mikrocontroller

Der Assistent darf zwischen der Übertragung vom Taschenrechner an den Computer und der Übertragung vom Computer an den Mikrocontroller nicht beendet werden.

1. Kompilieren des Programms
   (a) Anpassen des Programmcodes
   Der Assistent überträgt die Formeldatei in die Datei file.c und schreibt die Größe der Formeldatei in die Datei file.h und main.h
   (b) Kompilierung
   Die Formeldatei wird einschließlich des Programms für den Mikrocontroller kompiliert
2. Übertragung des Programms an den Mikrocontroller
   (a) Anschluss des Mikrocontrollers
   Der Assistent bittet den Nutzer den Mikrocontroller anzuschließen und den Anschluss mit Enter zu bestätigen
   (b) Reset
   Der Assistent bittet den Nutzer den Reset Taster auf der Platine zu betätigen, um den Bootloader zu starten
   (c) Übertragung
   Der Assistent überträgt die Datei an den Mikrocontroller, sobald der Bootloader startet

Der Assistent hat die Übertragung nun abgeschlossen und der Mikrocontroller sowie der Adapter können vom Computer getrennt werden.

7.4 Reset des Taschenrechners

Nach der Sicherung der Formel auf dem Mikrocontroller wird der Taschenrechner zurückgesetzt, um zu zeigen, dass die Manipulation funktioniert

1. Der Menüpunkt System (Taste G) wird im Hauptmenü aufgerufen
2. Der Menüpunkt Reset (F5) wird aufgerufen
3. Die nächste Seite wird aufgerufen (F6)
4. Der Menüpunkt “Alles zurücksetzen“ wird aufgerufen (F2)
5. Der Reset wird bestätigt (F1)
6. Der Taschenrechner wird neu gestartet (EXIT)
7. Die Grundeinstellungen werden durchgeführt

7.5 Wiederherstellung der Formel

1. Die Platine wird an den Taschenrechner angeschlossen
2. Der Reset Taster auf der Platine wird betätigt
3. Der Menüpunkt Link (Taste E) wird im Hauptmenü aufgerufen
4. Der Kabeltyp wird geändert (F4)
5. Der Kabeltyp wird auf 3PIN Kabel eingestellt (F2)
6. Die Übertragung startet automatisch
7. Die Erfolgsmeldung wird mit EXIT geschlossen

Die Formel steht nun im Formelspeicher wieder zur Verfügung

8 Zusammenfassung

Nach der Aufzeichnung der Kommunikation zweier Taschenrechner konnte das dabei verwendete Protokoll soweit nachvollzogen werden, dass der Empfang von Dateien durch einen Emulator möglich ist.

Aufbauend auf diesen Erkenntnissen wurde erfolgreich ein Modul entwickelt, dass leicht zu benutzen ist und als Speichererweiterung für den Taschenrechner gesehen werden kann, die es erlaubt beim Reset gelöschte Dateien wiederherzustellen.

9 Ausblick

9.1 Verbesserung des Moduls

Derzeit ist das Modul sehr groß und auffällig. Ein Lehrer könnte es in einer Prüfung leicht erkennen. Dies hängt jedoch hauptsächlich mit der verwendeten Fertigungstechnik zusammen. Die Platine wurde auf einer Lochrasterplatine aufgebaut und ist bedingt durch die bedrahteten Bauelemente sehr groß. Zudem ist bei Lochrasterplatinen nur eine sehr geringe Packungsdichte möglich.

Abhilfe schafft die Entwicklung einer SMD Platine. Dadurch, dass SMD Bauteile nicht durch die Platine durchgesteckt werden, sondern direkt auf der Oberseite der Platine verlötet werden, verbrauchen sie deutlich weniger Platz. Der verwendete Mikrocontroller ist in bedrahteter Bauform sehr groß. Leistungsstärkere Typen mit einer größeren Speicherkapazität in SMD Bauform sind dagegen deutlich kleiner. SMD Platinen sind nur noch wenige Millimeter dick und es reicht eine sehr geringe Fläche zur Unterbringung sämtlicher Bauteile aus.

Ein solches Modul ließe sich nach Betrachtung von Bildern eines geöffneten fx-CG 20[8] im Internet noch problemlos unterbringen. Es müssten 4 Drähte für Spannung, Daten und Masse angelötet werden. Es ist davon auszugehen, dass es einige Schüler gibt, die über entsprechende Fähigkeiten verfügen die Taschenrechner möglicherweise sogar für ihre Mitschüler mit einem entsprechenden Modul auszustatten.

In Zukunft könnte es ähnliche Module geben, wie für Spielekonsolen (Modchips), die fertig erworben werden können und nur noch eingebaut werden müssen.

Eine Speichererweiterung wäre zusätzlich durch den Einsatz einer SD Karte möglich, die einen Speicher von mehreren Gigabyte bereitstellen könnte, und somit deutlich mehr Speicherplatz böte als der in den Taschenrechner eingebaute Speicher.

Der verwendete Linearregler ist sehr ineffizient und für das Modul deutlich überdimensioniert. Ein anderer Spannungsregler kann sowohl den Strom- als auch den Platzverbrauch auf der Platine verringern. Alternativ ist es denkbar die benötigte Versorgungsspannung direkt von der Platine des Taschenrechners abzugreifen und gänzlich auf einen Spannungsregler zu verzichten.

9.2 Risiken für Schulen

Im Hinblick auf eine einheitliche Prüfungssituation wären solche Module ein enormes Problem. Es wird gerade im Abitur ein enormer Aufwand betrieben wie durch die Schule bereitgestelltes Papier oder teilweise mit Geräten, die eingeschaltete Handys erkennen sollen.

9.2.1 Rechtliche Bewertung

Der Nachweis des Einsatzes eines solchen Moduls dürfte sich als schwierig gestalten. Das Modul könnte so programmiert werden, dass es die Sicherheitskopie nur nach vorheriger Übertragung einer Datei mit einem Passwort an den Taschenrechner sendet. Da der Lehrer das Passwort nicht kennt, kann er nicht testen, ob ein Modul eine Sicherheitskopie an den Taschenrechner schickt.

Ein Lehrer könnte nach einer Prüfung den Formelspeicher des Taschenrechners durchsuchen, jedoch könnte der Schüler die betreffenden Daten bis dahin wieder gelöscht haben. Zudem ist es auch nicht verboten während einer Prüfung Daten im Taschenrechner zu speichern, weshalb der Schüler behaupten könnte, die Formel während der Prüfung eingespeichert zu haben.

Da die Taschenrechner in den meisten Fällen den Schülern gehören, ist ein Lehrer nicht berechtigt den Taschenrechner zu öffnen, um zu überprüfen, ob ein entsprechendes Modul eingebaut ist. Vor allem, da es beim Öffnen zum Verlust der Gewährleistung durch den Hersteller käme, ist ein generelles Öffnen aller Taschenrechner abgesehen von dem Zeitfaktor nicht möglich.

9.2.2 Maßnahmen für Schulen

Die einzige Möglichkeit sicherzustellen, dass keine manipulierten Geräte durch die Schüler verwendet werden ist, dass die Schule eigene Geräte für Prüfungen zur Verfügung stellt.

An Gymnasien würde dies bedeuten, dass für jeden Schüler in der Abschlussprüfung nach der 10. Klasse ein Prüfungsrechner bereitgestellt werden müsste. Bereits an kleinen Gymnasien schreiben dabei etwa 100 Schüler diese Prüfungen. Normalerweise liegt der Einzelpreis für einen fx-CG 20 bei etwa 110€. Selbst bei einem Mengenrabatt wird der Einzelpreis eines Rechners nicht unter 50€ fallen. Die Anschaffung einer ausreichenden Anzahl an Prüfungsrechnern würde an einem kleinen Gymnasium also mindestens 5000€ kosten. Auch die jährlichen Kosten für Batterien sind dabei nicht zu vernachlässigen

9.2.3 Maßnahmen für Hersteller

Hersteller dieser Taschenrechner können ihre Modelle zusätzlich absichern. Derzeit leicht zu öffnende Gehäuse könnten verklebt oder mit Spezialschrauben verschlossen werden. Transparente Gehäuse könnten den Einbau eines Moduls zur Manipulation sichtbar machen.

Auch die Abschaltung oder Entfernung der Kommunikationsschnittstellen (USB und 3PIN) könnte eine Schutzmaßnahme darstellen, jedoch werden damit viele Funktionen der Taschenrechner eingeschränkt.

Keine dieser Änderungen könnte zudem verpflichtend an derzeitigen Taschenrechnern der Schüler durchgeführt werden.

10 Erweiterte eigene Position

Ich war mir von Anfang an sicher, dass es möglich sein würde den Taschenrechner über die UART Schnittstelle zu manipulieren. Ursprünglich war es geplant nur Formeldateien speichern zu können. Meine Erwartungen wurden bei weiteren Tests übertroffen, und es stellte sich heraus, dass auch die Sicherung von anderen Dateitypen und sogar Backups des Taschenrechners möglich ist.

10.1 Erfahrungen

Die Entwicklung dieser Hardware war mit einigem Aufwand verbunden. Ich konnte dabei jedoch auch einiges lernen. Erstmalig stürzte ein Mikrocontroller aufgrund einer mangelhaft aufgebauten Schaltung ab. Im Laboraufbau auf einem Steckbrett gab es laufend Kontaktprobleme. Die Anfälligkeit von Mikrocontrollern steigt bei hohen Frequenzen deutlich.

Besonders wichtig war bei der Entwicklung, dass der Taschenrechner nicht beschädigt wird. Schließlich ist er sehr teuer und ich benötige ihn weiterhin in der Schule.

Erstmalig habe ich ein Projekt sehr ausführlich dokumentiert und Erfahrungen im Umgang mit dem Textsatzsystem LaTeX gesammelt. Für zukünftige Projekte werde ich diese Form der Dokumentation weiterverfolgen, jedoch werde ich ihren Umfang verringern.

Einer der wichtigsten Punkte bei dieser Manipulation ist in meinen Augen jedoch, dass kein Öffnen des Taschenrechners erforderlich ist. Selbst bei einem Garantiefall kann CASIO die Manipulation nicht nachweisen. Jedoch ist es zweifelhaft, ob eine externe Platine bei einer Klausur unentdeckt bleiben würde.

10.2 Fähigkeiten der Schüler

Die meisten Schüler werden nicht dazu in der Lage sein ein Modul zur Manipulation zu entwickeln. Jedoch dürfte der Einbau fertiger Module vielen Schülern möglich sein.

10.3 Weitere Taschenrechner

Auch wenn an dieser Stelle exemplarisch ein Modul zur Manipulation eines CASIO Taschenrechners entwickelt wurde, haben die Taschenrechner anderer Hersteller ähnliche Probleme. Auch die Grafikrechner der Firma Texas Instruments verfügen über eine der 3PIN Schnittstelle sehr ähnliche Kommunikationsmöglichkeit. Entsprechende Tests kann ich jedoch nicht durchführen.

Es sollte klar sein, dass es sich bei den hier vorgestellten Manipulationsmöglichkeiten um ein generelles Problem bei grafischen Taschenrechnern handelt.

11 Rechtliche Situation

11.1 Entwicklung eines Moduls zur Manipulation

Die Entwicklung eines solchen Moduls sehe ich als rechtlich unbedenklich an. Da ich meine Zustimmung zur EULA (Endbenutzer-Lizenzvereinbarung), die ich CASIO zum Testen verschiedener Software am Computer gegeben hatte vor Beginn der Entwicklung durch Deinstallation widerrufen habe, kommt ein Verstoß gegen das Verbot der Disassemblierung nicht in Frage. Generell stellt sich die Frage, ob ich dagegen verstoßen habe, schließlich habe ich zu keinem Zeitpunkt in das Betriebssystem des Taschenrechners eingegriffen.

11.2 Veröffentlichung von Details

Die Veröffentlichung von Details sehe ich kritisch. Insbesondere die Protokollmitschnitte könnten urheberrechtlich oder patentrechtlich geschützt sein.

Meine eigenen Schaltpläne sollte ich dagegen aus rechtlicher Sicht problemlos veröffentlichen dürfen, jedoch verzichte ich darauf, um Nachbauten zu vermeiden.

11.3 Verkauf eines Moduls zur Manipulation

Ob der Verkauf eines entsprechenden Moduls illegal ist, hängt in meinen Augen davon ab, inwiefern das Übertragungsprotokoll geschützt ist. Sollte dies geschützt sein, ist ein Verkauf vermutlich nicht zulässig.

Die reine Herstellung oder der Besitz eines solchen Moduls sollte insbesondere zum privaten Gebrauch keinesfalls illegal sein.

11.4 Verwendung in einer Klausur

Für die Verwendung in einer Klausur gilt, dass der Einsatz illegal ist und mit einem Täuschungsversuch gleichzusetzen ist.

12 Anhang

12.1 Datenblätter & Bedienungsanleitungen

12.1.1 Auszug Bedienungsanleitung[5] Seite 12

```
3-pin serial port
Method:
  Start-stop (asynchronous), half-duplex

Transmission speed (BPS):
  115200 bits/second (normal)
  9600 bits/second (When connected to CFX-9850G series or fx-7400G series
      calculator; Send/Receive commands)
  38400 bits/second (Send38k/Receive38k commands)

<115200 bits/second>
 Parity: EVEN
 Bit length: 8 bits
 Stop bit:
    Send: 1 bit
    Receive: 1 bit
    Includes parity (None) 1-bit
  X ON/X OFF Control: None

<9600, 38400 bits/second>
 Parity: None
 Bit length: 8 bits
 Stop bit:
    Send: 3 bits
    Receive: 2 bits
    Includes parity (None) 1-bit
  X ON/X OFF Control: None
```

12.2 Protokollmitschnitt

12.2.1 9600 Baud

Entfernt

12.2.2 115200 Baud

Entfernt

12.3 Ablauf der Kommunikation

12.3.1 9600 Baud

Entfernt

12.3.2 115200 Baud

Entfernt

12.4 Overhead

Entfernt

12.5 Abbildungen

12.5.1 Klinkenstecker

Entfernt

Abbildung 1: Klinkenstecker und Buchse des Taschenrechners

Entfernt

Abbildung 2: Nummerierung der Pins und Beschriftung der Kontakte des Klinkensteckers

12.5.2 Mitschnitt des Protokolls

Entfernt

Abbildung 3: Schaltplan der Platine zum Mitschnitt des Protokolls

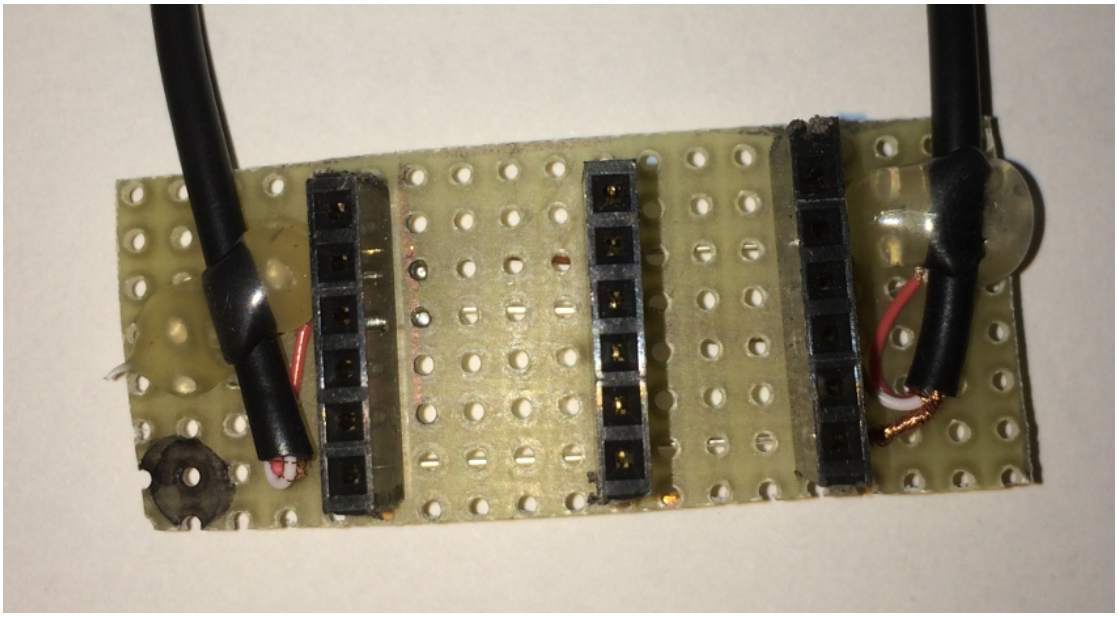

Abbildung 4: Foto der Platine zum Mitschnitt des Protokolls

12.6 Modul zur Manipulation

Entfernt

Abbildung 5: Schaltplan der Manipulationshardware

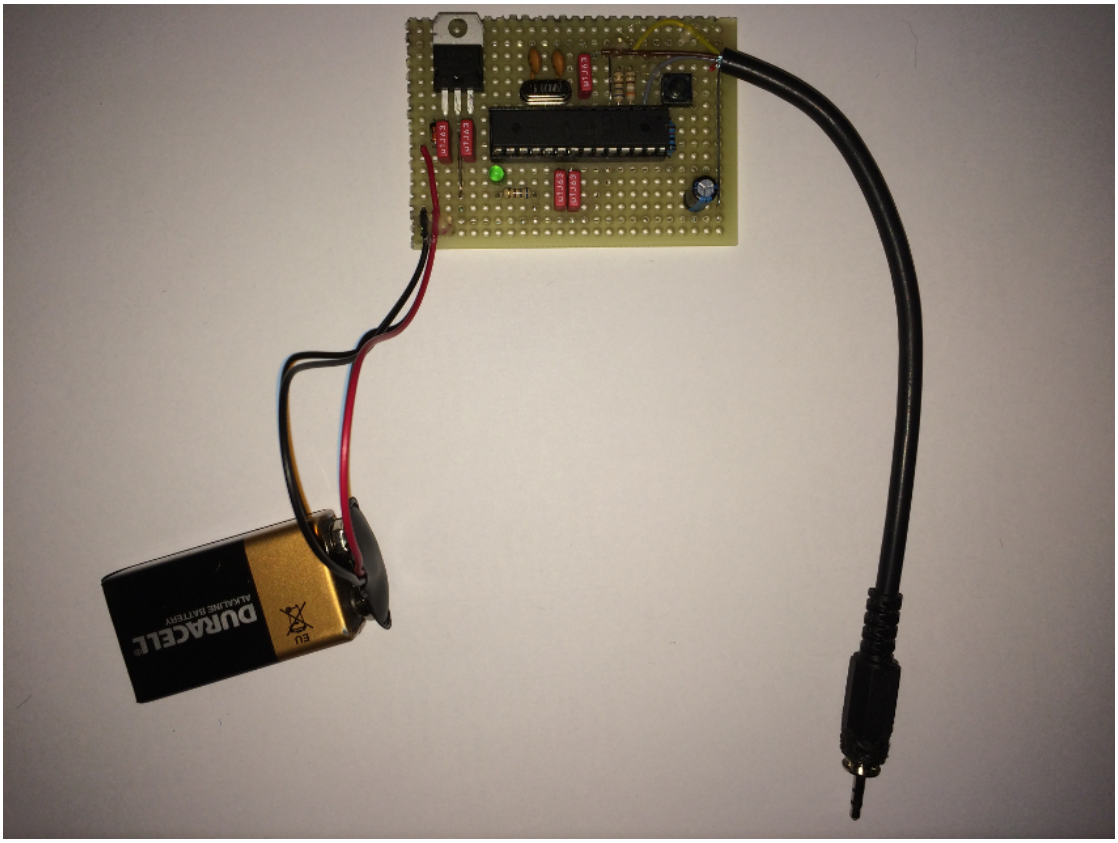

Abbildung 6: Manipulationsmodul

13 Literaturverzeichnis

[1] Ministerium für Schule und Weiterbildung NRW 2014: Digitale Alternativen für Mathe in der Oberstufe – Ministerin Löhrmann: Wir tragen dem technischen Fortschritt Rechnung http://www.schulministerium.nrw.de/docs/bp/Ministerium/Presse/Pressemitteilungen/2014_-16_-Legislaturperiode/PM20140410/pm_10_04_14-GTR-Erlass.pdf (10.07.2015)
verschoben nach: https://schulministerium.nrw.de/docs/bp/Ministerium/Presse/Pressemitteilungen/2014_16_LegPer/PM20140410/pm_10_04_14-GTR-Erlass.pdf (17.08.2015)
[2] CASIO 2013: FX-CG20 Betriebssystem-Update Version 2.00 http://www.casio-schulrechner.de/de/presse/detail/1383/ (10.07.2015)
[3] SimonLothar 2011: Re: Casio Prizm documentation https://www.omnimaga.org/casio-prizm/casio-prizm-documentation/msg207992/#msg207992 (10.07.2015)
[4] PrizmSDK Setup Guide http://prizm.cemetech.net/index.php?title=PrizmSDK_Setup_Guide (10.07.2015)
[5] CASIO: Hardware User Guide http://www.casioeducation.com/resource/manuals/PRIZM%20FX-CG10/Hardware_User_Guide_English.pdf (10.07.2015) (nicht mehr verfügbar)
[6] Peter Fleury: UART Library http://homepage.hispeed.ch/peterfleury/doxygen/avr-gcc-libraries/group__pfleury__uart.html (17.08.2015)
[7] fedel: AVR Hexfile Boot Loader http://sourceforge.net/projects/kavr/ (17.08.2015)
[8] prizm.cemetech.net: Pictures of the inside http://prizm.cemetech.net/index.php?title=Technical_Info#Pictures_of_the_inside (17.08.2015)

14 Impressum

Robin Meis
Zülpicherstr. 63
52388 Nörvenich
Telefon: +49 1523 4508223
Email: robin.meis@smartnoob.de
.
Webseite: http://blog.smartnoob.de
Für Rückfragen stehe ich gerne zur Verfügung

Alle verwendeten Firmen-, Markennamen und Warenzeichen sind Eigentum der jeweiligen Inhaber und dienen lediglich zur Identifikation und Beschreibung der Produkte und Dienstleistungen.